PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-022369
(43)Date of publication of application : 21.01.1997

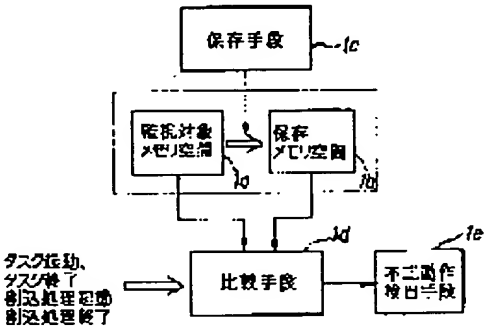
(51)Int.Cl. G06F 11/28
G06F 9/46
G06F 9/46

(21)Application number : 07-172020
(22)Date of filing : 07.07.1995

(71)Applicant : FUJITSU LTD
(72)Inventor : MASUDA HIDEJI

(54) ILLICIT OPERATION DETECTION METHOD IN KERNEL OF MULTI-TASKING SYSTEM

(57)Abstract:
PROBLEM TO BE SOLVED: To detect an illicit rewrite operation to a memory in terms of software and to search a task and a processing which performed the illicit rewrite operation.
SOLUTION: In this illicit operation detection method in the kernel of a multi-tasking system for executing a control transfer processing to a user interruption processing and a task dispatching processing for actuating a user task, a preservation means 1c preserves the contents of a monitoring object memory space 1a in a preservation memory space 1b beforehand, a comparison means 1d checks whether or not the memory contents of the monitoring object memory space 1a and the memory contents preserved in the memory 1b match with each other, as to the start of the user task, the processing end of the user task, the start of the user interruption processing and the end of user interruption processing, and an illicit operation detection means 1e discriminates a memory rewrite illicit operation by the user task or the user interruption processing based on the change of the memory contents.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平9－22369

(43)公開日　平成９年(1997)１月21日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　11/28		7313－5Ｂ	Ｇ０６Ｆ　11/28	Ａ	
9/46	３３０		9/46	３３０Ｃ	
	３４０			３４０Ａ	

審査請求　未請求　請求項の数12　ＯＬ　（全 17 頁）

(21)出願番号　特願平7－172020

(22)出願日　平成７年(1995)７月７日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　増田　秀二
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内

(74)代理人　弁理士　斉藤　千幹

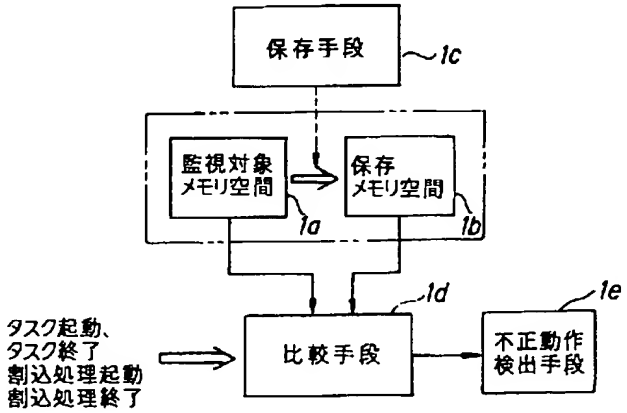
(54)【発明の名称】　マルチタスキング方式のカーネルにおける不正動作検出方法

(57)【要約】

【目的】　ソフト的にメモリへの不正な書替え動作を検出し、該不正書替え動作をしたタスク、処理を突き止める。

【構成】　利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、保存手段１ｃは予め監視対象メモリ空間１ａの内容を保存メモリ空間１ｂに保存し、比較手段１ｄは、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間１ａのメモリ内容とメモリ１ｂに保存してあるメモリ内容とが一致しているか調べ、不正動作検出手段１ｅはメモリ内容の変化に基づいて利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する。

本発明の第1の原理説明図



1

【特許請求の範囲】

【請求項１】　利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め設定されている監視対象メモリ空間の内容を保存する第１ステップ、

利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間のメモリ内容と保存してあるメモリ内容とが一致しているか調べる第２ステップ、

メモリ内容の変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する第３ステップを有することを特徴とする不正動作検出方法。

【請求項２】　前記第１ステップは、監視対象メモリ空間全体の内容を加算し、あるいはその他の演算処理を施して該演算結果を保存し、

第２ステップは、監視対象メモリ空間全体の内容を加算し、あるいはその他の演算処理を施し、該演算結果と保存してある演算結果を比較することにより監視対象メモリ空間のメモリ内容の変化を調べることを特徴とする請求項１記載の不正動作検出方法。

【請求項３】　前記方法は更に、デバッガを設け、該デバッガを介して監視対象メモリ空間を設定するステップを有し、

前記第３ステップは、

メモリ内容の変化が検出された時、デバッガを起動するステップ、

デバッガによりメモリ内容を書き替えた利用者タスクの識別子あるいは利用者割込み処理の識別子を出力するステップを有することを特徴とする請求項１記載の不正動作検出方法。

【請求項４】　利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め、監視対象である入出力装置のアドレスと、該入出力装置にアクセス可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格とをそれぞれ設定する第１ステップ、

利用者タスクあるいは利用者割込み処理からカーネルの有する入出力処理機能がコールされた時、該コール要求に含まれる入出力アドレスと前記設定されているアドレスを比較する第２ステップ、

一致している場合には、該コール要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格を比較する第３ステップ、

識別子とアクセス資格の不一致に基づいて、入出力装置に対する利用者タスクあるいは利用者割込み処理の不正アクセスを識別する第４ステップを有することを特徴とする不正動作検出方法。

2

【請求項５】　前記方法は、更に、デバッガを設け、該デバッガを介して前記監視対象である入出力装置のアドレスとアクセス資格を設定するステップを有し、

前記第４ステップは、

コール要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格の不一致が検出された時、デバッガを起動するステップ、

デバッガによりアクセスした利用者タスクの識別子あるいは利用者割込み処理の識別子を出力するステップを有することを特徴とする請求項４記載の不正動作検出方法。

【請求項６】　利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め、監視対象オブジェクトの識別子と該オブジェクトの操作が可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格とをそれぞれ設定する第１ステップ、

利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子と前記設定されているオブジェ識別子を比較する第２ステップ、

一致している場合には、該オブジェクト操作要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格を比較する第３ステップ、

利用者タスクあるいは利用者割込み処理の識別子とアクセス資格との不一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する第４ステップを有することを特徴とする不正動作検出方法。

【請求項７】　前記方法は、更に、デバッガを設け、該デバッガを介して前記監視対象オブジェクトの識別子とアクセス資格を設定するステップを備え、

前記第４ステップは、

オブジェクト操作要求を発行した利用者タスクあるいは利用者割込み処理の識別子とアクセス資格の不一致が検出された時、デバッガを起動するステップ、

デバッガによりアクセスした利用者タスクの識別子あるいは利用者割込み処理の識別子を出力するステップを有することを特徴とする請求項６記載の不正動作検出方法。

【請求項８】　利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め、監視対象オブジェクトの識別子及び該オブジェクトに対する操作内容をそれぞれ設定する第１ステップ、

利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子、操作内容が前記設定されているオブジェ識別子、操作内容とそれぞれ一致しているか比較する第２ステップ、

両者の一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する第３ステップを有することを特徴とする不正動作検出方法。

【請求項９】　前記方法は、更に、デバッガを設け、該デバッガを介して前記監視対象のオブジェクト識別子とオブジェクトに対する操作内容を設定するステップを備え、

前記第３ステップは、

前記両者の一致が検出された時、デバッガを起動するステップ、

デバッガによりアクセスした利用者タスクの識別子あるいは利用者割込み処理の識別子を出力するステップを有することを特徴とする請求項８記載の不正動作検出方法。

【請求項１０】　利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

予め、監視対象オブジェクトの識別子及びタスクの異常終了とみなす異常終了コードあるいは異常終了とみなさない異常終了コードをそれぞれ設定する第１ステップ、

利用者タスクあるいは利用者割込み処理からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、該システムコールに含まれるオブジェクト識別子が前記設定されているオブジェクト識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しているか比較する第２ステップ、

オブジェクト識別子及び異常終了コードの両方の一致に基づいて、あるいはオブジェクト識別子の一致、異常コードの不一致に基づいて利用者タスクあるいは利用者割込み処理の異常を検出する第３ステップを有することを特徴とする不正動作検出方法。

【請求項１１】　利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、

監視対象メモリにおける各空き領域の内容を保存する第１ステップ、

利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、各空き領域の内容と保存してある各空き領域の内容とが一致しているか調べる第２ステップ、

少なくとも１つの空き領域における内容変化に基づいて利用者タスク又は利用者割込み処理によるメモリ書替え

不正動作を識別する第３ステップを有することを特徴とする不正動作検出方法。

【請求項１２】　前記第１ステップは、監視対象メモリにおける空き領域毎に、空き領域の内容を加算し、あるいはその他の演算処理を施して、該演算結果を保存し、

第２ステップは、監視対象メモリにおける各空き領域毎に、空き領域の内容を加算し、あるいはその他の演算処理を施し、該演算結果と保存してある対応する空き領域の演算結果とを比較することにより監視対象メモリにおける各空き領域の内容が一致しているか調べることを特徴とする請求項１１記載の不正動作検出方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は不正動作検出方法に係わり、特に利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法に関する。近年の通信制御装置では組み込み型のＯ／Ｓ（オペレーティングシステム）を使用してソフトウェアで処理を行うのが常識となっている。このような通信装置では汎用ＣＰＵを使用し、Ｏ／Ｓはリアルタイム性や高速応答性が要求されることから、イベントドリブン型のマルチタスキング処理が可能なカーネルが主流である。組み込み型のＯ／Ｓにおいてもソフトウェアデバッグのためにデバッガが用意されているのが普通であり、デバッガの機能が充実していればソフト開発の効率化が可能となり、十分な試験を実施できることから装置の信頼性も向上する。本発明は、マルチタスク方式で最も厄介なバグの一つである、処理に無関係なタスクによるメモリの不正な書き替え等の検出方式に関するものであり、ディスパッチ処理において不正な動作の検出を行うため、特別なハードウェア機構を必要としない点が特徴である。

【０００２】

【従来の技術】マルチタスキング処理においては、複数のタスクが同時に動いて処理を実行する。かかるマルチタスキングによるデバッグ中に、Ｏ／Ｓにより共通に管理されている資源、例えばメモリの内容が不正に書き替えられることがある。かかる場合には、メモリを書き替えたタスクを発見する必要があるが、多数のタスクが並行して動作しているためソフト的にその発見をするのが難しかった。すなわち、不正にメモリの書き替えが行われた場合、書き替えられたアドレスに対するアクセスをソフトのみで検出する手段はなかった。このため、従来は、ＣＰＵのバス状態を監視するハードウェアを設け、該ハードウェアにより特定アドレスのアクセスを検出し、ＮＭＩ割り込み等でソフトに通知する機構を設けて特定アドレスに対してアクセスが行われたことを検出している。

【０００３】Ｏ／Ｓが管理するオブジェクト（セマフ

5

ォ、メールボックス、イベントフラグ等）に対する不正操作についても同様であり、ソフト的にオブジェクトに不正操作したタスクを発見するのが難しかった。このため、オブジェクトを管理しているメモリに対して前述のハードウェアにより不正な書き込みを行っている利用者タスクや利用者割込み処理を突き止める方法でデバッグを行っている。尚、システムコール・トレース機能を有するＯ／Ｓの場合には、トレース情報の解析により不正な操作を行っている処理を見つけることが行われている。

【０００４】

【発明が解決しようとする課題】しかし、上記従来の方法では以下の問題点がある。すなわち、

ハードウェアによるメモリアクセス検出法は、メモリ不正書替え動作を行ったタスクや処理を発見するための周辺回路が複雑で高価になる問題がある。

又、ハードウェアによるメモリアクセス検出法は、メモリプールの空き領域のようにダイナミックに変化する領域の不正書き込みに対処できない問題がある。これは、メモリプールの場合空き領域のアドレスが変化し、予め監視すべき領域（アドレス）を特定できないからである。

更に、ハードウェアによるメモリアクセス検出法では、オブジェクトに対する正当なアクセスも検出してしまう問題がある。すなわち、所定のオブジェクトに対する操作が許されている利用者タスクや利用者割込み処理からの正当なアクセスをも検出する。このため、アクセス資格との連携による不正アクセスのみの検出が要望されている。

不正なオブジェクト操作をトレース情報で検出する方法は、バグの発生状況によっては必要なトレース情報が失われて（トレース領域サイズ不足により）、原因が究明できない問題がある。

又、トレース情報が多いと不正な操作を行っている部分を見つけるのに手間がかかる問題がある。

【０００５】以上から本発明の第１の目的は、メモリへの不正な書替え動作をソフト的に検出できる不正動作検出方法を提供することである。本発明の第２の目的は、メモリプールの空き領域のようにダイナミックに変化するメモリ領域の不正書替え動作を検出できる不正動作検出方法を提供することである。本発明の第３の目的は、入出力装置に対する不正のアクセス動作を検出できる不正動作検出方法を提供することである。本発明の第４の目的は、アクセス資格との連携によりオブジェクトに対する不正な操作を検出できる不正動作検出方法を提供することである。

【０００６】

【課題を解決するための手段】前記第１の目的は本発明によれば、図１に示すように、予め設定されている監視対象メモリ空間１ａの内容を別のメモリ空間１ｂに保存

6

する手段１ｃと、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間１ａのメモリ内容とメモリ空間１ｂに保存してあるメモリ内容とが一致しているか調べる手段１ｄと、メモリ内容の変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する手段１ｅとにより達成される。・・・請求項１

【０００７】前記第２の目的は、本発明によれば、監視対象メモリにおける各空き領域の内容を保存する手段と、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、各空き領域の内容と保存してある各空き領域の内容とが一致しているか調べる手段と、少なくとも１つの空き領域における内容変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する手段とにより達成される。・・・請求項１１

【０００８】前記第３の目的は、本発明によれば、図２に示すように、予め、監視対象である入出力装置のアドレスＡＩＯと、該入出力装置にアクセス可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格ＡＱＲとをそれぞれ設定する手段２ａと、利用者タスクあるいは利用者割込み処理からカーネルの有する入出力処理機能がコールされた時、該コール要求に含まれる入出力アドレスと前記設定されているアドレスとを比較する第１の比較手段２ｂと、一致している場合には、該コール要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格ＡＱＲとを比較する第２の比較手段２ｃと、識別子とアクセス資格の不一致に基づいて、入出力装置に対する利用者タスクあるいは利用者割込み処理の不正アクセスを識別する手段２ｄとにより達成される。・・・請求項４

【０００９】前記第４の目的は、本発明によれば、図３に示すように、予め、監視対象オブジェクトの識別子ＩＤＯと該オブジェクトの操作が可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格ＡＱＲとをそれぞれ設定する手段３ａと、利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子と前記設定されているオブジェ識別子ＩＤＯとを比較する第１の比較手段３ｂと、一致している場合には、該オブジェクト操作要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格ＡＱＲとを比較する第２の比較手段３ｃと、利用者タスクあるいは利用者割込み処理の識別子とアクセス資格との不一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する手段３ｄとにより達成される。・・・請求項６

【００１０】又、前記第４の目的は、本発明によれば、

予め、監視対象オブジェクトの識別子及び該オブジェクトに対する操作内容をそれぞれ設定する手段と、利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子、操作内容が前記設定されているオブジェ識別子、操作内容とそれぞれ一致しているか比較する手段と、両者の一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する手段とにより達成される。・・・請求項８

【００１１】更に、前記第４の目的は、本発明によれば、予め、監視対象オブジェクトの識別子及びタスクの異常終了とみなす異常終了コードあるいは異常終了とみなさない異常終了コードをそれぞれ設定する手段と、利用者タスクあるいは利用者割込み処理からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、該システムコールに含まれるオブジェクト識別子が前記設定されているオブジェクト識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しているか比較する手段と、オブジェクト識別子及び異常終了コードの両方の一致に基づいて、あるいはオブジェクト識別子の一致、異常コードの不一致に基づいて利用者タスクあるいは利用者割込み処理の異常を検出する手段とにより達成される。・・・請求項１０

【００１２】

【作用】利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、保存手段１ｃ（図１）は予め監視対象メモリ空間１ａの内容を保存メモリ空間１ｂに保存し、比較手段１ｄは、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間１ａのメモリ内容とメモリ１ｂに保存してあるメモリ内容とが一致しているか調べ、不正動作検出手段１ｅはメモリ内容の変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する。この場合、保存手段１ｃは、監視対象メモリ空間１ａ全体の内容を加算し，演算結果を保存メモリ空間１ｂに保存し、比較手段１ｄは、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、監視対象メモリ空間１ａの全体の内容を加算し、加算結果と保存してある加算結果とを比較することにより監視対象メモリ空間１ａのメモリ内容の変化を検出する。以上のようにすれば、保存メモリ空間のメモリ容量を少なくできる。・・・請求項１、２

【００１３】監視メモリ空間が、メモリプールのように空き領域が変化する場合には、各空き領域の内容を保存し、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、各空き領域の内容と保存してある各空き領域の内容とが一致しているか調べ、少なくとも１つの空き領域における内容変化に基づいて所定の利用者タスク又は利用者割込み処理によるメモリ書替え不正動作を識別する。この場合、空き領域毎に空き領域の内容を加算し、加算結果を保存し、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、各空き領域毎に、該空き領域の内容を加算し、加算結果と保存してある対応する空き領域の加算結果とを比較することによりメモリ内容の変化を検出する。・・・請求項１１、１２

【００１４】利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、設定手段２ａ（図２）は、予め、監視対象である入出力装置のアドレスＡＩＯと、該入出力装置にアクセス可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格ＡＱＲとをそれぞれ設定し、第１比較手段２ｂは、利用者タスクあるいは利用者割込み処理からカーネルの有する入出力処理機能がコールされた時、該コール要求に含まれる入出力アドレスと前記設定されているアドレスを比較し、第２比較手段２ｃは、アドレスが一致している場合には、該コール要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格ＡＱＲを比較し、不正動作検出部２ｄは、識別子とアクセス資格の不一致に基づいて、入出力装置に対する利用者タスクあるいは利用者割込み処理の不正アクセスを識別する。・・・請求項４

【００１５】利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行するマルチタスキング方式のカーネルにおける不正動作検出方法において、設定手段３ａ（図３）は、予め、監視対象オブジェクトの識別子ＩＤＯと該オブジェクトの操作が可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格ＡＱＲとをそれぞれ設定し、第１比較手段３ｂは、利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子と前記設定されているオブジェ識別子ＩＤＯを比較し、識別子が一致している場合には、第２比較手段３ｃは、該オブジェクト操作要求を発行した利用者タスクあるいは利用者割込み処理の識別子と前記アクセス資格ＡＱＲを比較し、不正動作検出手段３ｄは、利用者タスクあるいは利用者割込み処理の識別子とアクセス資格との不一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する。・・・請求項６

【００１６】利用者タスクを起動するタスクディスパッチ処理及び利用者割込み処理への制御転送処理を実行す

るマルチタスキング方式のカーネルにおける不正動作検出方法において、予め、監視対象オブジェクトの識別子及び該オブジェクトに対する操作内容をそれぞれ設定し、利用者タスクあるいは利用者割込み処理からカーネルにオブジェクト操作が要求された時、該操作要求に含まれるオブジェクト識別子、操作内容が前記設定されているオブジェ識別子、操作内容とそれぞれ一致しているか比較し、両者の一致に基づいて、設定されているオブジェクトに対する利用者タスクあるいは利用者割込み処理の不正操作を識別する。このようにしても、オブジェクトに対する不正操作を識別できる。・・・請求項８

【００１７】又、予め、監視対象オブジェクトの識別子及びタスクの異常終了とみなす異常終了コード（あるいは異常終了とみなさない異常終了コード）をそれぞれ設定し、利用者タスクあるいは利用者割込み処理からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、該システムコールに含まれるオブジェクト識別子が前記設定されているオブジェクト識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しているか比較し、オブジェクト識別子及び異常終了コードの両方の一致に基づいて（あるいはオブジェクト識別子の一致、異常コードの不一致に基づいて）利用者タスクあるいは利用者割込み処理の異常を検出する。・・・請求項１０

【００１８】

【実施例】

（Ａ）メモリ不正書替え動作検出

(a) メモリ不正書き替え動作検出タイミングの説明

図４は本発明のメモリの不正書き替え動作検出タイミングの説明図である。１１はカーネルであり、タスクディスパッチ処理部１１ａ、利用者割り込み処理への制御転送処理部１１ｂから構成される。１２，１３は利用者タスク、１５は利用者割込み処理、１６は監視対象メモリである。イベントドリブン型のタスクスケジューリングでは、システムコール発行や外部割り込みにより、利用者タスク１２，１３の切替えや利用者割り込み処理１５の実行が行われる。図４の例では次の順序で利用者タスクや利用者割込み処理が実行される。

【００１９】処理(a)において、利用者タスク１２からシステムコールＡが発行されて利用者タスク１３の起動が要求されると、タスクディスパッチ処理部１１ａはタスクスケジューリングアルゴリズムに従って利用者タスク１３を起動する。利用者タスク１３の処理(b)が終了すると、利用者タスク１３はシステムコールＢを発行してタスク終了をタスディスパッチ処理部１１ａに通知する。これにより、タスクディスパッチ処理部１１ａは利用者タスク１２を起動し、利用者タスク１２はシステムコールＡの直後から処理(c)を実行する。利用者タスク１２の処理(c)の実行中に割り込みが発生すると、制御処理転送部１１ｂは利用者割り込み処理１５を起動する。利用者割り込み処理１５の処理(d)が終了すると、利用者割り込み処理１５は割込み処理終了を制御転送処理部１１ｂに通知する。これにより、制御転送処理部１１ｂは割り込み元である利用者タスク１２を起動し、利用者タスク１２は割込み発生直後から処理(e)を実行する。

【００２０】以上から明らかなように、処理切替え毎にカーネル１１のタスクディスパッチ処理あるいは制御転送処理が介在することになる。そこで、カーネル１１の処理の切替えタイミングをメモリ不正書き込みの検出タイミングとし、該検出タイミング　～　で監視対象メモリ１６の内容変化を調べる。処理(a)～(d)のどこかでメモリの書き替えが行われれば、監視対象メモリの内容が変化し、書き替えたタスクや処理を検出できる。図４の例では、検出タイミング　の時点のメモリ内容と検出タイミング　の時点のメモリ内容が不一致となり、利用者タスク１３の処理(b)の中で書き替えが行われたことが分かる。

【００２１】(b) メモリの不正書替え動作検出の第１実施例

図５は、利用者タスクによるメモリの不正書替え動作検出の第１実施例説明図である。図中、１１ａはカーネルにおけるディスパッチ処理部、１２～１４は利用者タスク、１６は監視対象メモリ、１７はメモリ監視範囲の内容を複写される作業領域、１８はメモリ監視範囲を記憶する監視範囲記憶部、１９はデバッガ、２０はコンソール装置である。

【００２２】カーネルにより利用者タスク１２が起動すると、(1) 利用者タスク１２はデバッガ強制起動割込みを発生し、デバッガ１９を起動する。デバッガ１９は強制起動割り込みにより起動されると、コンソール装置２０を利用したコマンド入力が可能となる。(2) かかる状態において、コンソール装置２０よりコマンド受け付け部１９ａを介してメモリ監視範囲が入力されると、デバッガ１９はメモリ監視設定コマンド１９ｂを実行して入力されたメモリ監視範囲を記憶部１８に設定し、(3) 同時に、指定範囲のメモリ内容を監視対象メモリ１６から作業領域１７に複写する。そして、デバッガ終了コマンドにより、利用者タスク１２に復帰する。(4) 利用者タスク１２よりタスク切替のシステムコールが発生すると、(5) ディスパッチ処理部１１ａは、監視対象メモリ１６と作業領域１７の内容を比較する。(6) 比較の結果、メモリ内容が一致していれば、ディスパッチ処理部１１ａは利用者タスク１３を起動してタスクを切り替える。これにより、利用者タスク１３の処理が開始する。

【００２３】(7) 利用者タスク１３よりタスク切替のシステムコールが発生すると、(8) タスクディスパッチ処理部１１ａは前記と同様に監視対象メモリ１６と作業領域１７の内容を比較する。(9) メモリ内容が一致していれば、利用者タスク１４にタスクが切り替わる。(10)

しかし、不一致であれば、タスクディスパッチ処理部１１ａは、利用者タスク１４への切替を行わず、デバッガ１９を起動する。デバッガ１９が起動されることにより、利用者はコンソール装置２０を利用して不正なメモリ書き込みを行った利用者タスクと書き込みを行ったプログラムの範囲を突き止めることができる。すなわち、カーネルは、不正動作が検出された時点（メモリ内容不一致検出の時点）で、該不正動作を行ったタスク識別子と検出部位（プログラム範囲）をログ収集して保持し、デバッガ１９はコンソール装置２０からの不正タスク表示要求により不正タスク表示コマンド１９ｃを実行し、カーネルからログ収集結果を受信してコンソール装置２０のディスプレイ画面に表示する。以上では、利用者タスクによる不正書替え動作検出を説明したが、利用者割込み処理による不正書替え動作検出も同様に行われる。

【００２４】(c) メモリの不正書替え動作検出の第２実施例

図６は、利用者タスクによるメモリの不正書替え動作検出の第２実施例説明図である。図中、１１ａはカーネルにおけるディスパッチ処理部、１２～１４は利用者タスク、１６は監視対象メモリ、１７′はメモリ監視範囲の全内容を加算した加算結果を記憶する加算結果格納領域、１８はメモリ監視範囲を記憶する監視範囲記憶部、１９はデバッガ、２０はコンソール装置である。第１実施例の場合、メモリ内容の比較において監視範囲が広いと作業領域１７の確保が難しくなる。そこで、メモリ内容の変化を検出するための別の手段として、監視範囲内のメモリ内容の加算を行い、加算結果の比較によるメモリ内容変化検出法を用いる。第２実施例では、書き替えられた内容によっては変化を検出できない場合があるが、非常に稀にしか発生せず、メモリ不正書き込みの検出の目的には十分実用になる。

【００２５】カーネル１１により利用者タスク１２が起動すると、(1) 利用者タスク１２はデバッガ強制起動割込みを発生し、デバッガ１９を起動する。デバッガ１９は強制起動割り込みにより起動されると、コンソール装置２０を利用したコマンド入力が可能となる。(2) かかる状態において、コンソール装置２０よりコマンド受け付け部１９ａを介してメモリ監視範囲が入力されると、デバッガ１９はメモリ監視設定コマンド１９ｂを実行してメモリ監視範囲を記憶部１８に設定し、(3)同時に、指定範囲のメモリ内容を加算し、加算結果を加算結果記憶領域１７′に格納する。しかる後、デバッガ終了コマンドにより、利用者タスク１２に復帰し、該利用者タスクが実行される。(4) 利用者タスク１２よりタスク切替のシステムコールが発生すると、(5) ディスパッチ処理部１１ａは、監視対象メモリ１６におけるメモリ監視範囲の内容を加算し、(6) 該加算結果と加算結果格納領域１７′に格納されている加算結果を比較する。(7) 比較の結果、加算結果が一致していれば、ディスパッチ処理部１１ａは利用者タスク１３を起動してタスクを切り替える。(8) 次に利用者タスク１３よりシステムコールの発行によりタスク切替え要求が発生すると、(9) タスクディスパッチ処理部１１ａは同様に監視対象メモリ１６におけるメモリ監視範囲の内容を加算し、(10) 該加算結果と加算結果格納領域１７′に格納されている加算結果を比較する。(11) 比較の結果、加算結果が一致していれば、利用者タスク１４に切り替わる。(12) しかし、不一致であれば、タスクディスパッチ処理部１１ａは、利用者タスク１４への切替を行わず、デバッガ１９を起動する。デバッガ１９が起動されることにより、利用者は第１実施例の場合と同様にコンソール装置２０を利用して不正なメモリ書替えを行った利用者タスクと書替えを行ったプログラムの範囲を突き止めることができる。以上では、加算を施した場合について説明したが、減算、排他的論理和演算その他の演算を施すことができる。

【００２６】（B）入出力装置に対す不正アクセス検出

図７は入出力装置に対する不正アクセス検出の実施例である。図中、１１はカーネルであり、ディスパッチ処理部１１ａ、入出力処理部１１ｃを有している。１２～１４は利用者タスク、１９はデバッガ、２０はコンソール装置、２１は記憶部であり、コンソール装置よりデバッガを介して設定された監視対象の入出力装置のアドレスＡＩＯと、該入出力装置にアクセス可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格ＡＱＲとを記憶する。カーネルの入出力処理機能が利用者タスクからＣＡＬＬされると、同時に入出力装置のアドレスが通知されるので、このアドレスと設定されている監視対象の入力装置のアドレスＡＩＯの比較する。一致すれば、アクセス資格ＡＱＲとカーネル１１が認識している現在動作中の利用者タスク、利用者割込み処理等の識別子を比較する。一致しなければ、アクセス資格のないタスクあるいは処理から監視対象の入出力装置にアクセスが行われたことを検出できる。

【００２７】以下、入出力装置に対す不正アクセス検出について詳細に説明する。カーネル１１により利用者タスク１２が起動すると、(1) 利用者タスク１２はデバッガ強制起動割込みを発生し、デバッガ１９を起動する。デバッガ１９は強制起動割り込みにより起動されると、コンソール装置２０を利用したコマンド入力が可能となる。(2) かかる状態において、コンソール装置２０よりコマンド受け付け部１９ａを介して監視対象である入出力装置のアドレスＡＩＯ（＝１００）、アクセス資格ＡＱＲ（＝利用者タスク１４の識別子）が入力されると、デバッガ１９は入出力監視設定コマンド１９ｄを実行してこれらＡＩＯ，ＡＱＲを記憶部２１に設定する。(3) そして、デバッガ終了コマンドにより、利用者タスク１２に復帰する。(4) 利用者タスク１２よりシステムコールにより利用者タスク１３の起動が要求されると、(5)

ディスパッチ処理部１１ａは、タスクスケジューリングアルゴリズムに従って利用者タスク１３を起動し、処理を開始する。(6) そして、利用者タスク１３が入出力アドレス＝１００として入出力処理機能をＣＡＬＬすると、(7) カーネル１１の入出力処理部１１ｃは、不正アクセス判定処理を行う。

【００２８】図８は不正アクセス判定処理のフローである。入出力処理機能がコールされると（ステップ１０１）、入出力処理部１１ｃはＣＡＬＬと同時に通知される入出力装置のアドレスと記憶部２１に設定されている監視対象のアドレスＡＩＯの比較を行う（ステップ１０２）。アドレスが一致してなければ、処理を継続し（ステップ１０３）、一致していれば、アクセス資格ＡＱＲとカーネル１１が認識している現在動作中の利用者タスクの識別子を比較する（ステップ１０４）。一致すれば、正当なアクセスである。正当なアクセスの場合には、入出力処理実行後、利用者タスク１３に復帰するので、不正なアクセスが発生しない限り利用者タスク１３の処理は通常通りに実行され続ける（ステップ１０３）。一方、一致しなければ、アクセス資格のないタスクあるいは処理から監視対象の入出力装置にアクセスが行われたことになり（不正アクセス）、利用者タスクの実行を中断してデバッガ１９を起動する（ステップ１０５、１０６）。

【００２９】図７の例では、記憶部２１に設定されているアクセス対象アドレスＡＩＯ（＝１００）と利用者タスク１３から依頼された入出力アドレス（＝１００）と一致している。そこで、カーネル１１が管理している動作中の処理（利用者タスク１３）とアクセス資格（＝利用者タスク１４）を比較する。(8) 一致していないため,入出力処理部１１ｃは不正なアクセスと判断して、利用者タスクの実行を中断しデバッガを起動する。デバッガが起動されることにより、利用者はコンソール装置２０を利用して不正動作を行ったタスクや、入出力処理をＣＡＬＬしたプログラムを突き止めることができる。すなわち、カーネル１１は、不正アクセス動作が検出された時点で、該不正アクセスを行ったタスクの識別子と検出部位（プログラム範囲）をログ収集して保持し、デバッガ１９はコンソール装置２０からの不正タスク表示要求によりカーネルからログ収集結果を受信してコンソール装置２０のディスプレイ画面に表示する。

【００３０】（Ｃ）オブジェクトに対する不正操作検出

(a) オブジェクトに対する不正操作検出の第１実施例

図９はオブジェクトに対する不正操作検出の第１実施例の構成図である。図中、１１はカーネルであり、ディスパッチ処理部１１ａ、オブジェクト操作処理部１１ｄを有している。１２～１４は利用者タスク、１９はデバッガ、２０はコンソール装置、２１は記憶部であり、コンソール装置よりデバッガを介して設定された監視対象オブジェクトの識別子ＩＤＯと、該オブジェクト操作が可能な利用者タスクあるいは利用者割込み処理を特定するアクセス資格ＡＱＲとを記憶する。オブジェクト不正操作検出の概略は以下の通りである。例えば、利用者タスク１３からオブジェクト操作処理機能が呼出されると、同時にオブジェクト識別子が通知されるので、オブジェクト操作処理部１１ｄはこの識別子と設定されている監視対象オブジェクトの識別子ＩＤＯを比較する。一致すれば、設定されているアクセス資格ＡＱＲとカーネルが認識している現在動作中の利用者タスクの識別子あるいは利用者割込み処理の識別子とを比較する。一致しなければ、アクセス資格のないタスクあるいは処理が監視対象オブジェクトを不正に操作したことを検出できる。

【００３１】オブジェクトとしてはセマフォ、メールボックス、イベントフラグ、メモリプール等があり、Ｏ／Ｓにより管理されている。このうちセマフォは、整数を値としてとる変数で、Ｐ及びＶ命令によってのみ値が変更される。１つのセマフォに対しては、一時的にはただ１つのタスクだけがＰ又はＶ命令を実行できる。Ｓをセマフォ値とするとき、あるタスクがＰ（Ｓ）を実行すると、Ｓの値が１減らされた後、そのＳの値が

Ｓ≧０なら、そのタスクは実行を続ける。

Ｓ＜０なら、そのタスクは停止し、Ｓ対する待ち行列に入る。

タスクＶ（Ｓ）を実行すると、Ｓの値が１増加された後、そのＳの値が

Ｓ＞０なら、そのタスクは実行を続ける。

Ｓ≦０なら、Ｓに対する待ち行列から１つタスクを取り出し、そのタスクの実行を再開させる。Ｖ（Ｓ）を実行したタスク自身も実行を継続する。

【００３２】図１０はＲＭＳ ６８Ｋのセマフォを用いた排他制御の説明図である。尚、ATSENは、そのタスクがある危険部分の排他制御のためにセマフォを用いることを宣言するもの、WTSENはセマフォを通して、危険部分の使用を要求するものであり、SGSEM はその使用を開放するものである。又、WTSENはセマフォ値を－１し、SGSENは＋１する。まずタスクｂがセマフォを通して危険部分の使用を要求する。この場合、セマフォ値が１なので、その要求は受け付けられ、使用が許可される。そのとき、セマフォ値は－１されて０となる。しかる後、タスクａが走行し、同じセマフォに対して使用要求したとき、セマフォ値が１でないので、使用中であり、使用が許可されず、タスクａはセマフォキューに入る。さらに、タスクｃが走行し、同じセマフォを要求するが、同様にセマフォキューにタスクｃも入る。この時のセマフォ値は２回－１され、－２になっている。その後、タスクｂがSGSENにより使用を開放し、セマフォキューにあるタスクａが、その使用を許可される。更に、タスクａが開放することにより、タスクｃの使用が許可され、最後にセマフォ値は１に戻り、セマフォキューも空となる。又、メールボックスはタスク間の交信手段であり、

15

あるタスクによってメッセージが置かれ、他のタスクによってメッセージが取り出される構造で表現された待ち行列である。メッセージを送る側のタスクはメールボックスに対してメッセージを送り、又、メッセージを受信する側のタスクはメールボックス上でメッセージの到着を待つ。かかるメールボックスを用いて同期制御や排他制御が行われる。

【００３３】以下、図９に従ってオブジェクトの不正操作検出方法を詳述する。カーネル１１により利用者タスク１２が起動すると、(1) 利用者タスク１２はデバッガ強制起動割込みを発生し、デバッガ１９を起動する。デバッガ１９は強制起動割り込みにより起動されると、コンソール装置２０を利用したコマンド入力が可能となる。(2) かかる状態において、コンソール装置２０よりコマンド受け付け部１９ａを介して監視対象オブジェクトの識別子ＩＤＯ（＝２８）、アクセス資格ＡＱＲ（＝利用者タスク１４の識別子）が入力されると、デバッガ１９はオブジェクト監視設定コマンド１９ｅを実行してこれらＩＤＯ，ＡＱＲを記憶部２１に設定する。(3) しかる後、デバッガ終了コマンドにより、利用者タスク１２に復帰する。(4) 利用者タスク１２よりシステムコールにより利用者タスク１３の起動が要求されると、(5) ディスパッチ処理部１１ａは、タスクスケジューリングアルゴリズムに従って利用者タスク１３を起動し、利用者タスク１３は処理を開始する。(6) そして、利用者タスク１３がシステムコール(オブジェクト識別子＝２３）を発行してオブジェクト操作を要求すると、(7) カーネル１１のオブジェクト操作処理部１１ｄは、不正操作判定処理を行う。

【００３４】図１１は不正操作判定処理のフローである。オブジェクト操作のシステムコールが発行されると（ステップ２０１）、オブジェクト操作処理部１１ｄはシステムコールに含まれるオブジェクト識別子と記憶部２１に設定されているオブジェクト識別子ＩＤＯとの比較を行う（ステップ２０２）。オブジェクト識別子が一致してなければ、処理を継続し（ステップ２０３）、一致していれば、アクセス資格ＡＱＲとカーネル１１が認識している現在動作中の利用者タスクの識別子を比較する（ステップ２０４）。一致すれば、正当なオブジェクト操作である。正当なオブジェクト操作の場合には、オブジェクト操作処理後、利用者タスク１３に復帰するので、不正なオブジェクト操作が発生しない限り利用者タスク１３の処理は通常通りに実行され続ける（ステップ２０３）。一方、一致しなければ、アクセス資格のないタスクあるいは処理が監視対象のオブジェクトを操作したことになり（不正操作）、利用者タスクの実行を中断してデバッガ１９を起動する（ステップ２０５、２０６）。

【００３５】図９の例では、記憶部２１に設定されているオブジェクト識別子ＩＤＯ（＝２８）と利用者タスク１３が発行したシステムコールに含まれるオブジェクト識別子（＝２８）が一致している。そこで、カーネル１１が管理している動作中の処理（＝利用者タスク１３）とアクセス資格（＝利用者タスク１４）を比較する。(8) 一致していないため、オブジェクト操作処理部１１ｄは不正なオブジェクト操作と判断して、利用者タスクの実行を中断しデバッガを起動する。デバッガ１９が起動されることにより、利用者はコンソール装置２０を利用して不正操作を行ったタスクやオブジェクト操作のシステムコールを発行したプログラム部分を突き止めることができる。すなわち、カーネル１１は、オブジェクトの不正操作が検出された時点で、該不正操作を行ったタスクの識別子と検出部位（プログラム範囲）をログ収集して保持し、デバッガ１９はコンソール装置２０から不正タスク表示要求が入力されるとカーネルからログ収集結果を受信してコンソール装置２０のディスプレイ画面に表示する。

【００３６】(b) 第２実施例

図１２はオブジェクト不正操作の検出の第２実施例の構成図である。１１はカーネル、１１ｅはイベント通知処理部、１２は利用者カーソル、１９はデバッガ、２０はコンソール装置、２１はオブジェクト操作検出条件が設定される記憶部、２３はイベントフラグ管理情報記憶部である。第２実施例では、イベントフラグの不正操作を例にして説明する。又、フラグＩＤ＝１０のフラグに対し、本来セットされるはずのないビット７がセットされる現象が発生しているものとする。

【００３７】カーネル１１により利用者タスク１２が起動すると、(1) 利用者タスク１２はデバッガ強制起動割込みを発生し、デバッガ１９を起動する。デバッガ１９は強制起動割り込みにより起動されると、コンソール装置２０を利用したコマンド入力が可能となる。(2) かかる状態において、コンソール装置２０よりコマンド受け付け部１９ａを介して監視対象オブジェクトの識別子ＩＤＯ（＝１０）、機能ＦＮＣ（＝セットフラグ）、データＤＴＡ（＝セットビット７）が入力されると、デバッガ１９はオブジェクト操作条件設定コマンド１９ｆを実行してこれらＩＤＯ，ＦＮＣ，ＤＴＡを記憶部２１に設定する。(3) そして、デバッガ終了コマンドにより、利用者タスク１２の実行を再開させる。(4) 利用者タスク１２がイベントフラグ操作のシステムコール(フラグＩＤ＝１０、セットフラグ、セットビット７）を発行してイベントフラグ操作を要求する。(5) カーネルのイベント通知処理部１１ｅは、セットフラグのシステムコールが発行されると、デバッガの設定条件をもとに不正操作検出処理を実行する。

【００３８】図１３はイベントフラグの不正操作検出の処理フローである。オブジェクト操作（セットフラグ）のシステムコールが発行されると（ステップ３０１）、イベント通知処理部１１ｅは、記憶部２１に設定されて

16

いるオブジェクト操作条件（イベントフラグ操作条件）を参照する（ステップ３０２）。ついで、システムコールに含まれるオブジェクト操作条件と設定されているオブジェクト操作検出条件との比較を行う（ステップ３０３）。すなわち、システムコールに含まれるオブジェクト識別子（フラグＩＤ＝１０）、機能コード（＝セットフラグ）、データ（＝セットビット７）と、記憶部２１に設定されているオブジェクト識別子ＩＤＯ（＝１０）、機能コードＦＮＣ（＝セットフラグ）、データ（＝セットビット７）とをそれぞれ比較する。

【００３９】オブジェクト操作条件が異なれば、正当なオブジェクト操作であり、イベントフラグ操作処理後に利用者タスク１３に復帰し、利用者タスク１３の処理を継続する（ステップ３０４，３０５）。一方、オブジェクト操作条件が一致すれば、利用者タスク１２が設定されているイベントフラグの不正操作をしたことになり（不正操作）、利用者タスクの実行を中断してデバッガ１９を起動する（ステップ３０６，３０７）。図１２の例では、オブジェクト操作検出条件が一致するから、イベント通知処理部１１ｅは利用者タスク１２の実行を中断し、デバッガ１９を起動することになる。デバッガが起動されることにより、利用者はコンソール装置２０を利用して不正操作を行ったタスクやシステムコールを発行したプログラム範囲を突き止めることができる。

【００４０】(c) オブジェクト不正操作検出の第３実施例

図１４はオブジェクト操作のシステムコールが単純に異常終了したことによりタスクの異常検出を行う第３実施例の構成図である。図中、１１はカーネル、１１ｆはシステムコール処理部、１２は利用者タスク、１９はデバッガ、２１はオブジェクト操作検出条件が設定される記憶部である。デバッガ１９は既に説明したように強制起動により起動する。そして、(1) デバッガ１９はオブジェクト操作条件設定コマンドに従ってオブジェクト操作検出条件を記憶部２１に設定する。第３実施例は、オブジェクト操作のシステムコールが、単純に異常終了するとタスクに異常ありと判定するものである。このため、オブジェクト操作検出条件におけるオブジェクトの種類／オブジェクト識別子として「全種類」が、システムコール異常終了コードとして「全種類」がそれぞれ設定される。

【００４１】デバッガによるオブジェクト操作検出条件設定が終了すれば、利用者タスク１２の処理が開始する。そして、(2) 利用者タスク１２がオブジェクト操作のシステムコールを発行して所定のオブジェクト操作を要求すると、(3) カーネル１１のシステムコール処理部１１ｆは要求されたオブジェクト操作を実行しようとする。しかし、要求されたオブジェクト操作が実行できない場合、例えば、セマフォを獲得したいけれどセマフォがない、あるいは、領域を獲得したいけれど領域がない等により要求されたオブジェクト操作が実行できない場合には、システムコール異常終了とする。(4) かかる異常終了により、システムコール処理部１１ｆは記憶部２１に設定されているオブジェクト操作検出条件を参照する。

【００４２】オブジェクト操作検出条件において、オブジェクトの種類／オブジェクト識別子は「全種類」、システムコール異常終了コードは「全種類」となっている。このため、システムコール処理部１１ｆは、システムコールで要求されたオブジェクトの種類や異常終了コードに関係なく、直ちに利用者タスク１２の実行を中断し、デバッガ１９を起動する。以後、利用者はコンソール装置２０を利用して異常終了したタスクを突き止めることができる。

【００４３】(d) オブジェクト不正操作検出の第４，第５実施例

図１５は、指定したオブジェクト操作が異常終了したことによりタスクの異常を検出する第４実施例の構成図である。第４実施例において、第３実施例（図１４）と異なる点は、　オブジェクト種別／オブジェクト識別子を指定すると共に、　タスクの異常とみなす異常終了コードを指定している点である。予め、監視対象オブジェクトの種類／識別子(イベントフラグ/識別子＝１０）及びタスクの異常とみなす異常終了コード（コード＝８）をそれぞれ記憶部２１に設定しておく。利用者タスク１２からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、イベント通知処理部１１ｅは、該システムコールに含まれるオブジェクトの種別／識別子が前記設定されているオブジェクトの種別／識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しているか比較する。オブジェクトの種類／識別子及び異常終了コードの両方が一致している場合には、イベント通知処理部１１ｅは、システムコールを発行した利用者タスク１２の異常とみなし、該タスクの実行を中断し、デバッガを起動する。以後、利用者はコンソール装置を利用して異常終了したタスクを突き止めることができる。

【００４４】図１６は、指定したオブジェクトが指定した異常終了コード以外の異常で終了したことによりタスクの異常を検出する第５実施例の構成図である。第５実施例において、第３実施例（図１４）と異なる点は、オブジェクト種別／オブジェクト識別子を指定すると共に、　タスクの異常とみなさない異常終了コードを指定している点である。予め、監視対象オブジェクトの種類／識別子(イベントフラグ/識別子＝１０）及びタスクの異常とみなさない異常終了コード（コード＝７）をそれぞれ記憶部２１に設定しておく。利用者タスク１２からのシステムコールにより要求されたオブジェクト操作に対する異常終了時、イベント通知処理部１１ｅは、該システムコールに含まれるオブジェクトの種類／識別子が

前記設定されているオブジェクトの種類／識別子と一致するか、及び、異常終了コードが前記設定されている異常終了コードと一致しないか比較する。オブジェクトの種類／識別子が一致し、異常終了コードが不一致の場合、イベント通知処理部１１ｅは、システムコールを発行した利用者タスク１２の異常とみなし、該タスクの実行を中断し、デバッガを起動する。以後、利用者はコンソール装置を利用して異常終了したタスクを突き止めることができる。

【００４５】（Ｄ）メモリプールの空き領域に対する不正書き込み動作の検出

図１７は、メモリプールの空き領域に対する不正書き込み動作検出方式の説明図であり、監視対象メモリはメモリプール内の空き領域で、内容変化検出方式は加算結果比較方式である。図中、３１はメモリプールで、左側はメモリブロック返却前のメモリプール、右側はメモリブロック返却後のメモリプールである。メモリプール３１の空き領域はメモリブロック返却によりダイナミックに変化する。従って、メモリブロック返却ごとにメモリプールの空き領域３１ａ～３１ｃの内容加算を行い、加算結果を空き領域の最終アドレス３１ａ′～３１ｃ′に最終ワードとして格納する。又、ディスパッチ処理部によるディスパッチ処理では、各空き領域毎に内容の加算を行い、加算結果と該空き領域の最終アドレスに格納されている加算結果とを比較し、一致／不一致により空き領域への不正書き込みの有無を判断する。

【００４６】図１８はメモリプールの空き領域に対する不正書き込み動作検出処理のフロー図である。タスク切換のシステムコールあるいは利用者割込み等が発生したか、換言すれば、検出タイミングになったかチェックする（ステップ４０１）。検出タイミングになれば、すなわち、利用者タスクの起動、利用者タスクの処理終了、利用者割込み処理の起動、利用者割込み処理終了のそれぞれにおいて、メモリプール３１の各空き領域毎に内容の加算を行い、加算結果と対応する空き領域の最終アドレスに格納されている加算結果とを比較する（ステップ４０２）。少なくとも１つの空き領域において加算結果が異なる場合には、空き領域への不正書き込みがあったものとみなし、利用者タスクの実行を中断し、デバッガを起動する（ステップ４０３、４０４）。以後、利用者はコンソール装置を利用してメモリプールの空き領域に不正書き込みをしたタスクあるいは処理を突き止めることができる。

【００４７】一方、ステップ４０２において、全空き領域の加算結果が一致（空き領域の内容が一致）していれば、あるいは、ステップ４０１において、「ＮＯ」であれば、メモリブロックの返却（リリースブロック）によりメモリプールの空き領域が変化したかチェックする（ステップ４０５）、変化してなければ初めに戻り、ステップ４０１以降の処理を繰り返す。しかし、空き領域

が変化すれば、新たな空き領域の内容を加算し（ステップ４０６）、該加算結果を空き領域の最終アドレスに格納し（ステップ４０７）、以後、初めに戻り、ステップ４０１以降の処理を繰り返す。尚、演算は加算に限らず、減算、排他的論理和演算その他の演算を採用することができる。又、以上では空き領域の加算結果を比較して空き領域への不正書き込み動作を検出したが、空き領域の内容を保存しておき、保存内容と空き領域の内容を比較することにより不正書き込み動作を検出することもできる。以上、本発明を実施例により説明したが、本発明は請求の範囲に記載した本発明の主旨に従い種々の変形が可能であり、本発明はこれらを排除するものではない。

【００４８】

【発明の効果】以上本発明は、異常となる現象は分かっているが、その現象を引き起こすタスクあるいは処理が不明な場合に適用して有効である。又、マルチタスキング方式のＯ／Ｓでは、タスクが数１０本同時に動作しているのが普通なので、異常現象だけ分かっていても発生元タスクを特定するのは困難なことが多い。本発明によれば、異常現象をブレーク条件として設定することにより、該現象が発生した時点で利用者タスクを中断してデバッガを起動するため、デバッガのメモリ表示機能等により異常現象の発生元タスクやシステムの状態を調査することができる。

【００４９】本発明によれば、ソフト的にメモリへの不正な書替え動作を検出し、該不正書替え動作をしたタスク、処理を突き止めることができる。この結果、デバッグを効率良く行うことができる。又、この場合、監視メモリ領域の内容変化、すなわち、監視メモリ領域への不正書替え動作を、該領域の内容を加算してその加算結果の比較により行うことにより、不正書替え動作検出に使用するメモリ容量を著しく少なくできる。本発明によれば、メモリプールの空き領域のようにダイナミックに変化するメモリ領域の不正書替え動作を検出し、該不正書替え動作をしたタスク、処理を突き止めることができる。この結果、デバッグを効率良く行うことができる。又、空き領域の内容変化を加算結果の比較により行うことにより、不正書替え動作検出に使用するメモリ容量を著しく少なくできる。

【００５０】本発明によれば、アクセス資格との連携により、入出力装置に対する不正のアクセス動作やオブジェクトに対する不正なアクセス操作を検出し、該不正アクセス、不正操作をしたタスク、処理を突き止めることができる。この結果、デバッグを効率良く行うことができる。

【図面の簡単な説明】

【図１】本発明の第１の原理説明図である。

【図２】本発明の第２の原理説明図である。

【図３】本発明の第３の原理説明図である。

21

【図４】メモリ不正書替え検出タイミングの説明図である。

【図５】利用者タスクによるメモリの不正書替え動作検出の第１実施例である。

【図６】利用者タスクによるメモリの不正書替え動作検出の第２実施例である。

【図７】入出力装置に対する不正アクセス動作検出の実施例である。

【図８】Ｉ／Ｏ不正アクセスの判定処理フローである。

【図９】オブジェクトに対する不正操作検出の第１実施例である。

【図１０】セマフォを用いた排他制御の説明図である。

【図１１】オブジェクト不正操作の判定処理フローである。

【図１２】オブジェクト不正操作検出の第２実施例である。

【図１３】イベントフラグ不正操作検出の処理フローである。

22

【図１４】オブジェクト不正操作検出の第３実施例である。

【図１５】オブジェクト不正操作検出の第４実施例である。

【図１６】オブジェクト不正操作検出の第５実施例である。

【図１７】メモリプールの空き領域に対する不正書替え検出の説明図である。

【図１８】メモリプールの空き領域に対する不正書き込み動作検出の処理フローである。

【符号の説明】

１ａ・・監視対象メモリ空間

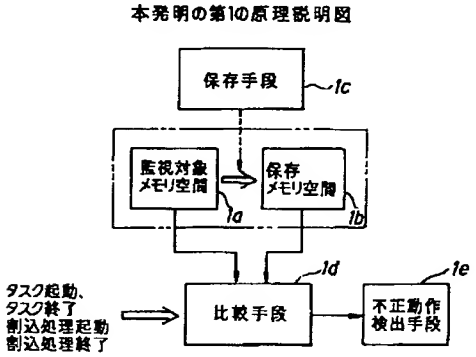
１ｂ・・保存メモリ空間

１ｃ・・保存手段

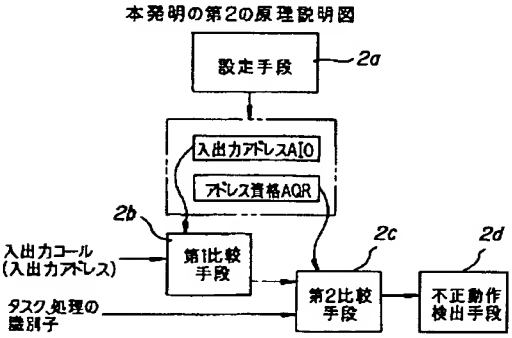
１ｄ・・比較手段

１ｅ・・不正動作検出手段

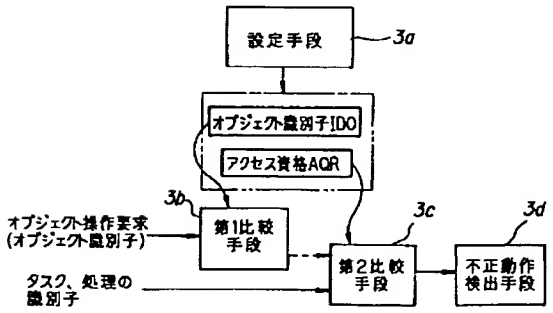
【図１】



【図２】

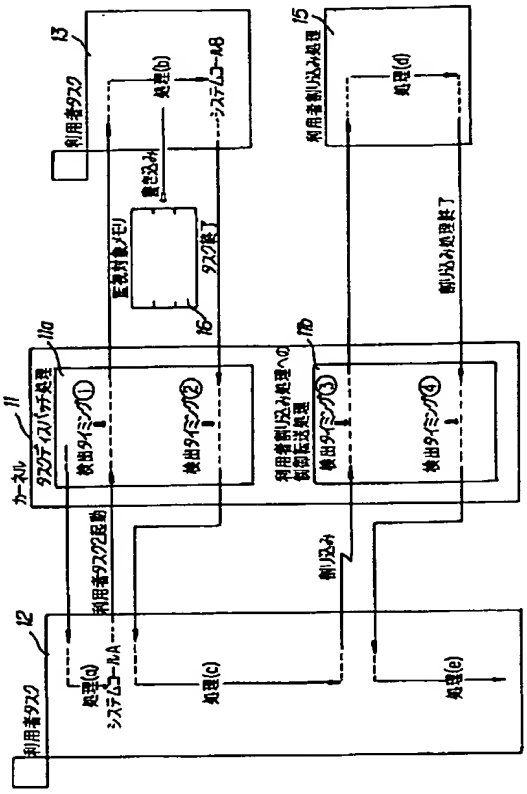


【図３】



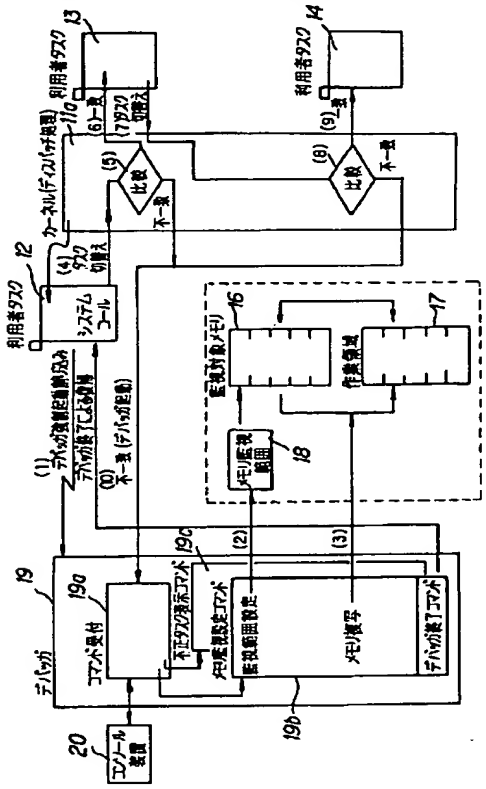
【図４】

メモリ書き替えの検出タイミング説明図



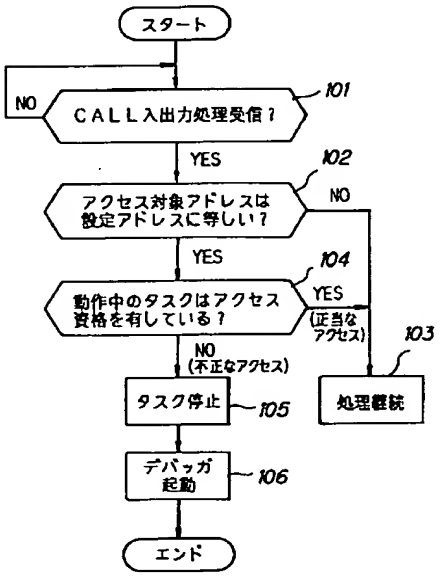
【図５】

利用者タスクによるメモリの不正書き替え動作検出の第1実施例



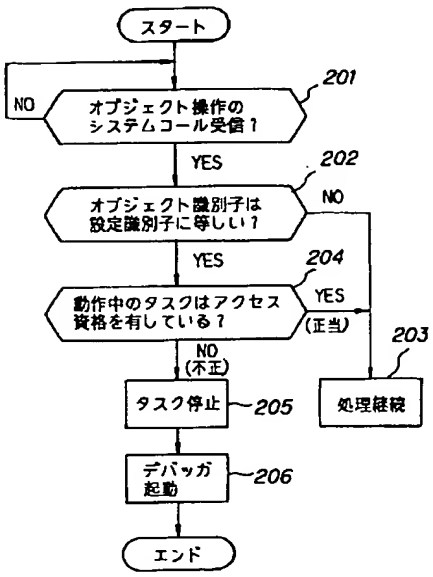
【図８】

I／O 不正アクセス判定処理



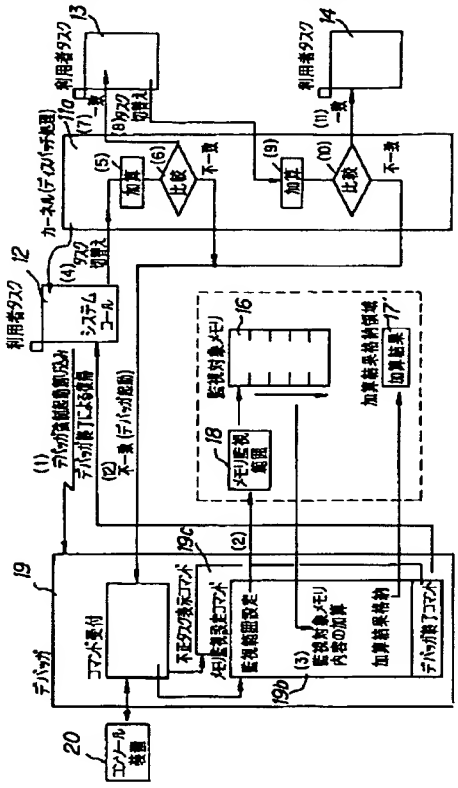
【図１１】

オブジェクト不正操作の判定処理



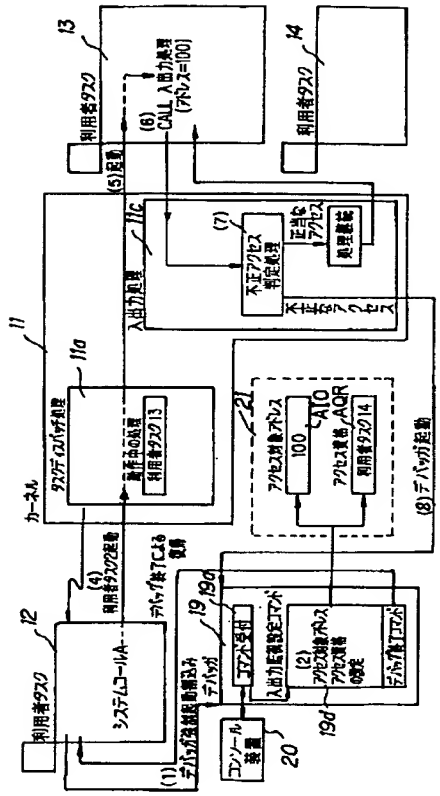
【図６】

利用者タスクによるメモリの不正書き替え動作検出の第2実施例



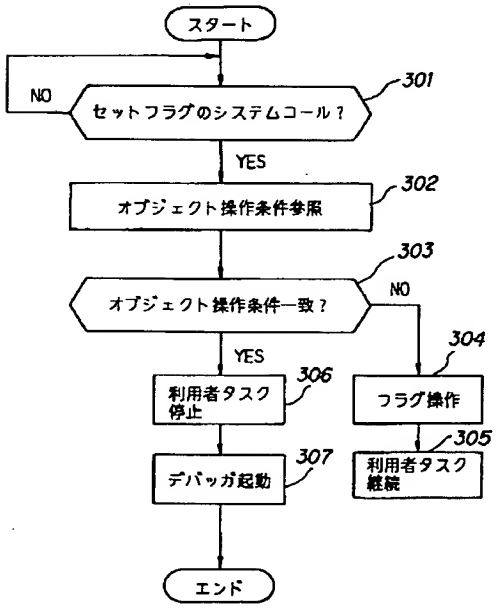
【図７】

入出力装置に対する不正アクセス動作検出の実施例



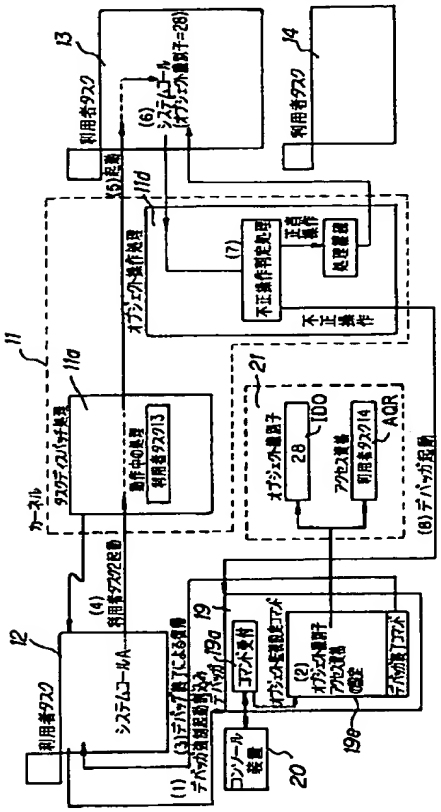
【図１３】

イベントフラグ不正操作検出処理



【図９】

オブジェクトに対する不正操作検出の第1実施例



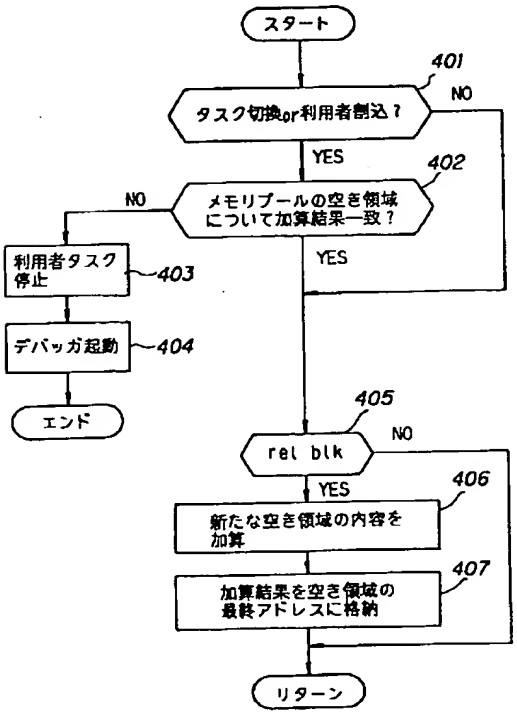
【図１０】

セマフォを用いた排他制御の説明

	タスクa	タスクb	タスクc	セマフォ値	セマフォキュー
初期				1	空
t_1		ATSEM WTSEM		0	↓
		使用		↓	↓
t_2	ATSEM WTSEM	↓		−1	タスクa
t_3		↓	ATSEM WTSEM	−2	タスクa タスクc
		↓	↓	↓	↓
t_4		SGSEM	↓	−1	タスクc
	使用	↓	↓	↓	↓
t_5	SGSEM	↓	↓	0	↓
	↓	↓	使用	↓	空
	↓	↓	↓	↓	↓
	↓	↓	SGSEM	1	↓
	↓	↓	↓	1	空

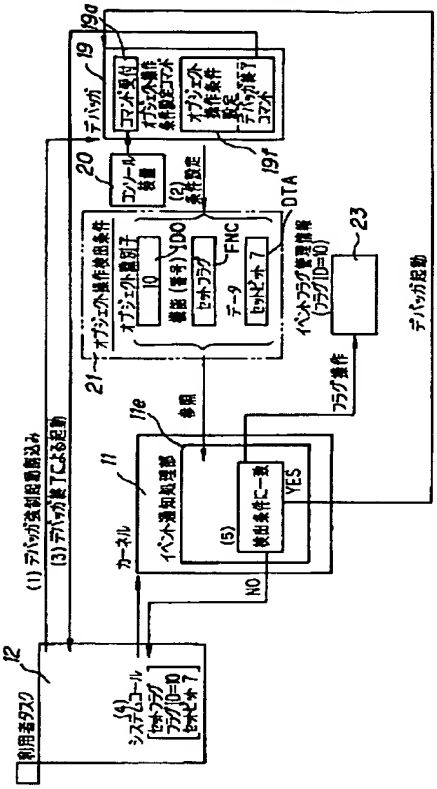
【図１８】

メモリプールの空き領域に対する不正書き込み
動作検出処理のフロー



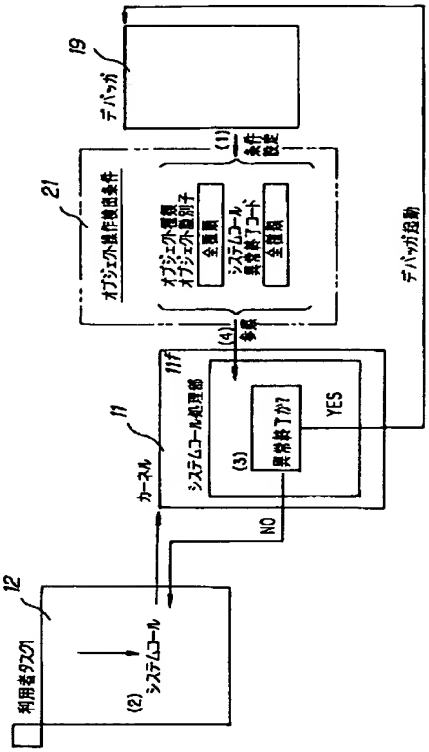
【図１２】

オブジェクト不正操作検出の第2実施例



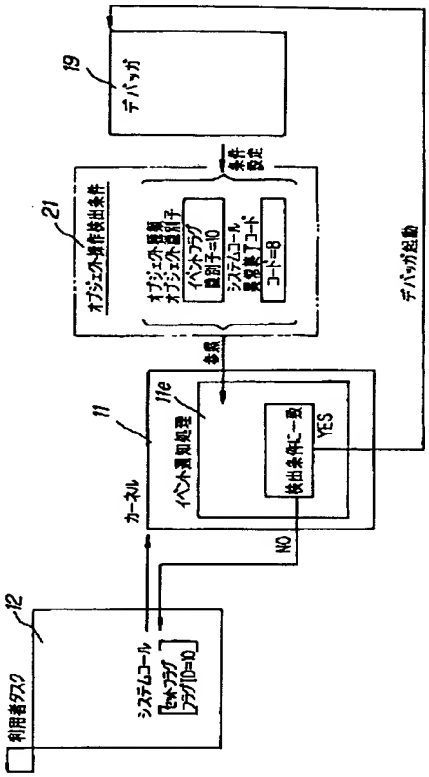
【図１４】

オブジェクト不正操作検出の第3実施例



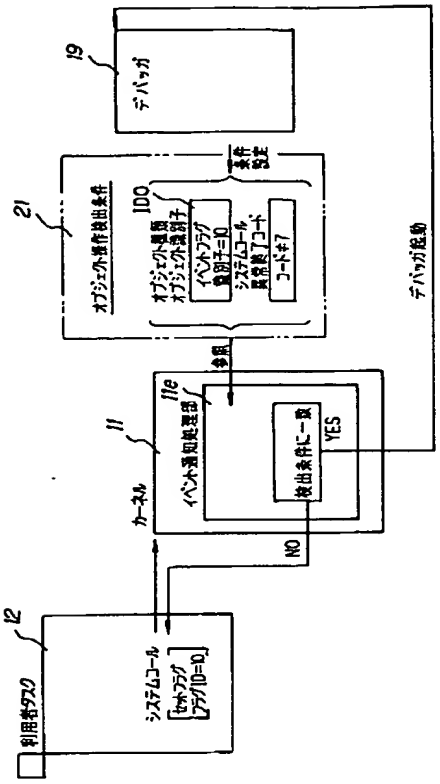
【図１５】

オブジェクト不正操作検出の第4実施例



【図１６】

オブジェクト不正操作検出の第5実施例



【図１７】

メモリプールの空き領域に対する不正書き替え検出説明図